WEST Search History

Hide Items | Restore | Clear | Cancel

DATE: Thursday, September 23, 2004

Hide?	Set Name	Query	Hit Count
	DB=PGPB,USPT,USOC; PLUR=YES; OP=ADJ		
☐	L14	L13 and L3	14
☐	L13	L5	76
	DB=EPAB,JPAB,DWPI,TDBD; PLUR=YES; OP=ADJ		
☐	L12	L5	0
	DB=PGPB,USPT,USOC,EPAB,JPAB,DWPI,TDBD; PLUR=YES; OP=ADJ		
☐	L11	Single instruction same multi?branch	0
☐	L10	Single instruction same multiple branch	8
☐	L9	Single instruction same multiple jump	1
☐	L8	Single instruction same multi?jump	0
☐	L7	Single instruction same multi?loop	0
☐	L6	Single instruction same multiple loop	1
☐	L5	L4 and multiple loop	76
☐	L4	Single instruction	7164
	DB=PGPB,USPT,USOC; PLUR=YES; OP=ADJ		
☐	L3	L1 or L2	3477
☐	L2	712/10-24.ccls.	1972
☐	L1	717/119,131,149-161.ccls.	1553

END OF SEARCH HISTORY

CiteSeer Find: [ ] Documents | Citations

Searching for **single instruction and multiple and loop**.
Restrict to: Header Title Order by: Expected citations Hubs Usage Date Try: Google (CiteSeer) Google (Web) CSB DBLP
93 documents found. **Order: number of citations.**

IMPACT: An Architectural Framework for.. - Chang, Mahlke.. (1991) (Correct) (107 citations)
achieve solid speedup over high-performance **single-instruction**-issue processors. We ran experiments to
1991 266 IMPACT: An Architectural Framework for **Multiple**-Instruction-Issue Processors Pohua P. Chang
function inline expansion, instruction placement, **loop** unrolling, **loop** peeling, memory disambiguation,
ftp.crhc.uiuc.edu/pub/IMPACT/conference/isca-91-framework.ps

One or more of the query terms is very common - only partial results have been returned. Try Google (CiteSeer).

A New Method for Mapping Optimization Problems onto Neural.. - Peterson, Söderberg (1989) (Correct) (92 citations)
[4] for the TSP case (see below)3 SIMD=**Single Instruction Multiple** Data. Figure 2: A K =4 graph
a or not. We denote this encoding scheme neuron **multiple**xing. Needless to say this encoding is not as
that the set connections constitute a closed **loop** (see fig. 3) and an extension in the sense that
ftp.thep.lu.se/pub/Preprints/89/lu_tp_89_01.ps.gz

Beyond Multiprocessing - Multithreading the SunOS Kernel - Eykholt, Kleiman.. (1992) (Correct) (78 citations)
in the current thread structure with a **single instruction**. The current LWP, process, and CPU
applications as a structuring technique to manage **multiple** asynchronous activities the kernel benefits
This removes various diversions in the idle **loop** and trap code and replaces them with
sunsite.unc.edu/pub/sun-info/development-tools/multi-threaded/beyond_mp.ps

The Superthreaded Architecture: Thread Pipelining with Run-time.. - Tsai, Yew (1996) (Correct) (67 citations)
blocks need to be grouped together in a **single instruction** stream. As a larger instruction window size
Abstract This paper presents a new concurrent **multiple**threaded architectural model, called
the superthreaded architectural model can exploit **loop**-level parallelism from a broad range of
www-users.cs.umn.edu/Research/Agassiz/Paper/tsai.pact96.ps.Z

Compiling Fortran D for MIMD Distributed-Memory Machines - Hiranandani (1992) (Correct) (55 citations)
Parallel Computer Forum (PCF) Fortran [24]**Single-instruction**, **multiple**-data (SIMD) machines such as the
the architecture of the underlying machine. **Multiple**-instruction, **multiple**data MIMD) shared-memory
with explicit synchronization and parallel **loops** found in Parallel Computer Forum (PCF) Fortran
www.cs.umd.edu/~keleher/papers/fortrand.ps.gz

Massively Parallel Methods for Engineering and Science.. - Camp Plimpton Hendrickson (1994) (Correct) (43 citations)
parallel architectures may be SIMD (**single-instruction**, **multiple**-data) or MIMD
we discuss the advantages a message-passing **multiple**-instruction/**multiple**-data (MIMD) programming
paths to memory. These fine-grained (or inner-**loop**) uses of parallelism are combined with
www.cs.sandia.gov/~sjplimp/papers/cacm94.ps.gz

Zero-Cycle Loads: Microarchitecture Support for Reducing Load.. - Austin (1995) (Correct) (33 citations)
levels of the data memory hierarchy in a **single instruction**. A significant body of work is dedicated to
decode, and another for pipelines with **multiple** decode stages. We evaluate these designs in a
eliminates the entire load operation, or (**loop**) blocking eliminates many cache miss latencies. In
ftp.cs.wisc.edu/sohi/papers/1995/micro.zcl.ps.gz

The M-Machine Multicomputer - Fillo, Keckler, Dally, Carter.. (1995) (Correct) (22 citations)
one for each ALU. All operations in a **single instruction** issue together but may complete out of
units, on-chip cache, and local memory. The **multiple** function units are used to exploit both
parallelized by identifying tasks, such as **loop** iterations, that can be distributed both across
cva.stanford.edu/pub/publications/AI1532.ps.Z

Radix Sort For Vector Multiprocessors - Zagha (1991) (Correct) (21 citations)
previously designed for a highly parallel **Single Instruction Multiple** Data (SIMD) computer, the

designed for a highly parallel **Single Instruction Multiple** Data (SIMD) computer, the Connection Machine allowing the algorithm to be fully vectorized. **Loop** Raking: A new technique we call **loop** raking is www.cs.cmu.edu/afs/cs.cmu.edu/project/scandal/public/papers/cray-sort-supercomputing91.ps.gz

Memory Access Coalescing: A Technique for Eliminating.. - Davidson, Jinturkar (1994) (Correct) (20 citations)
This gathering of dependencies into a **single instruction** can adversely affect instruction of newer machines with wide-buses to load/ store **multiple** floating-point operands in a single memory memory references for possible coalescing, **loops** are sometimes unrolled by the optimizer. However, ftp.cs.virginia.edu/pub/techreports/CS-93-62.ps.Z

Partitioned Register File for TTAs - Janssen, Corporaal (1996) (Correct) (18 citations)
Instead of packing the operations in a **single instruction**, like VLIWs, TTAs pack **multiple** transports (ILP) should be able to read and write **multiple** register values concurrently ILP architectures execution, controlling function inlining, **loop** unrolling and to compute spilling costs during cardit.et.tudelft.nl/MOVE/papers/micro28.ps

Exploiting Basic Block Value Locality with Block Reuse - Jian Huang (1998) (Correct) (16 citations)
All of these schemes work at the level of a **single instruction**, and try to predict the next value that approach, while the MOAC incorporates **multiple** separate processors on a single chip. The the GCC compiler. Compiler optimizations, such as **loop**-unrolling and function inlining, affect the sizes www.cs.umn.edu/Research/Agassiz/Paper/huangj.hpca5.ps.gz

Relaxing SIMD Control Flow Constraints using Loop.. - von Hanxleden, Kennedy (1992) (Correct) (16 citations)
loop nest causes special problems for SIMD (**Single Instruction**, **Multiple** Data) architectures because of on a shared-memory or distributed-memory MIMD (**Multiple** Instruction, **Multiple** Data) machine. However, Relaxing SIMD Control Flow Constraints Using **Loop** Transformations Reinhard von Hanxleden Ken softlib.rice.edu/pub/CRPC-TRs/reports/CRPC-TR92207-S.ps.gz

MISC: A Multiple Instruction Stream Computer - Tyson (1992) (Correct) (15 citations)
and provides an alternative approach to **single instruction** stream multi-issue machines such as # MISC: A **Multiple** Instruction Stream Computer Gary Tyson Andrew R. between the instructions with the data dependency. **Loop** unrolling is a compiler optimization that can american.cs.ucdavis.edu/publications/MISC.tech.ps

The Superthreaded Processor Architecture - Jenn-Yuan Tsai (1999) (Correct) (13 citations)
from different basic blocks in a **single instruction** stream need to be examined and issued architecture (CMA) that can exploit the **multiple** granularities of parallelism available in serious when a compiler attempts to pipeline a **loop** with many conditional branches [21]In www.cs.umn.edu/Research/Agassiz/Paper/tsai.ieee.ps.gz

Automatic Generation of DAG Parallelism - Ron Cytron (1989) (Correct) (13 citations)
constructs imply sequential execution: a **single instruction** stream exists before, during, and after constructs that enable the execution of **multiple** concurrently-executed instruction streams: doall relying on a datadependence -based and **loop**-based representation as input [Kuc78] Wol82] www.mcs.newpaltz.edu/~hind/papers/dag.ps

Data Prefetching And Data Forwarding In Shared Memory.. - Poulsen, Yew (1994) (Correct) (13 citations)
Write operations. A Forwarding Write is a **single instruction** that combines a write and a cycle. Forwarding Write instructions that specify **multiple** destination processors issue **multiple** parallel numerical application codes with **loop**-level and vector parallelism. More data, www.cs.umn.edu/Research/Agassiz/Paper/poulsen.icpp94.ps.Z

Exploiting a New Level of DLP in Multimedia Applications. - Corbal, Valero, Espasa (1999) (Correct) (11 citations)
together with more recent SIMD-like (**Single Instruction Multiple** Data) ISAs (such as MMX)we have with more recent SIMD-like (**Single Instruction Multiple** Data) ISAs (such as MMX)we have developed a parallelism found in any single parallel level (**loop**)Additionally,we will discuss int dist1(blk1, www.ac.upc.es/homes/jcorbal/82.ps

Comparing Static And Dynamic Code Scheduling for.. - Chang, Chen, Mahlke, Hwu (1991) (Correct) (9 citations)
two times speedup over a high performance **single-instruction**-issue processor. However, the performance Comparing Static And Dynamic Code Scheduling for **Multiple**-Instruction-Issue Processors Pohua P. Chang

been mostly based on numerical applications and **loop** kernels. The results presented in this paper are
ftp.crhc.uiuc.edu/pub/IMPACT/conference/micro-91-dynamic.ps

First 20 documents Next 20

Try your query at: Google (CiteSeer) Google (Web) CSB DBLP

Terms used **single instruction** and **multiple loop** Found **611** of **142,983**

Sort results by: relevance

Display results: expanded form

Save results to a Binder

Search Tips

☐ Open results in a new window

Try an Advanced Search

Try this search in The ACM Guide

Results 1 - 20 of 200 Result page: **1** 2 3 4 5 6 7 8 9 10 next

Best 200 shown Relevance scale

1 Efficient distributed event-driven simulations of multiple-loop networks

B. D. Lubachevsky

February 1989 **Communications of the ACM**, Volume 32 Issue 1

Full text available: pdf(1.97 MB) Additional Information: full citation, abstract, references, citings, index terms, review

Simulating asynchronous multiple-loop networks is commonly considered a difficult task for parallel programming. Two examples of asynchronous multiple-loop networks are presented in this article: a stylized queuing system and an Ising model. In both cases, the network is an n × n grid on a torus and includes at least an order of n2 feedback loops. A new distributed simulation algorithm is demonstrated on these two examples. The algorithm combines three elements: (...

2 Emerging areas: A new look at exploiting data parallelism in embedded systems

Hillery C. Hunter, Jaime H. Moreno

October 2003 **Proceedings of the international conference on Compilers, architectures and synthesis for embedded systems**

Full text available: pdf(322.12 KB) Additional Information: full citation, abstract, references, index terms

This paper describes and evaluates three architectural methods for accomplishing data parallel computation in a programmable embedded system. Comparisons are made between the well-studied *Very Long Instruction Word (VLIW)* and *Single Instruction Multiple Packed Data (SIMpD)* paradigms; the less-common *Single Instruction Multiple Disjoint Data (SIMdD)* architecture is described and evaluated. A taxonomy is defined for data-level parallel archi ...

Keywords: DLP, DSP, ILP, SIMD, VLIW, architecture, data-level parallelism, embedded, media, processor, sub-word parallelism, telecommunications

3 Speculative multithreaded processors

Pedro Marcuello, Antonio González, Jordi Tubella

July 1998 **Proceedings of the 12th international conference on Supercomputing**

Full text available: pdf(1.24 MB) Additional Information: full citation, references, citings, index terms

Keywords: control speculation, data dependence speculation, data speculation, dynamically

scheduled processors, multithreaded processors

4 Synchronous relaxation for parallel simulations with applications to circuit-switched networks

Stephen G. Eick, Albert G. Greenberg, Boris D. Lubachevsky, Alan Weiss

October 1993 **ACM Transactions on Modeling and Computer Simulation (TOMACS)**, Volume 3 Issue 4

Full text available: pdf(1.86 MB) Additional Information: full citation, abstract, references, citings, index terms, review

Synchronous relaxation, a new, general-purpose, efficient method for parallel simulation, is proposed. The method is applied to obtain a new parallel algorithm for simulating large circuit-switched communication networks. To show that synchronous-relaxation method is efficient, we present the results of circuit-switched network simulation experiments, and analytic approximations derived from a mathematical model of the simulation method.

Keywords: discrete event, fixed point, parallel, relaxation

5 Multi-threaded vectorization

Tzi-cker Chiueh

April 1991 **ACM SIGARCH Computer Architecture News , Proceedings of the 18th annual international symposium on Computer architecture**, Volume 19 Issue 3

Full text available: pdf(1.06 MB) Additional Information: full citation, references, citings, index terms

6 Unboundedly parallel simulations via recurrence relations

Albert G. Greenberg, Boris D. Lubachevsky, Isi Mitrani

April 1990 **ACM SIGMETRICS Performance Evaluation Review , Proceedings of the 1990 ACM SIGMETRICS conference on Measurement and modeling of computer systems**, Volume 18 Issue 1

Full text available: pdf(1.36 MB) Additional Information: full citation, abstract, references, citings, index terms

New methods are presented for parallel simulation of discrete event systems that, when applicable, can usefully employ a number of processors much larger than the number of objects in the system being simulated. Abandoning the distributed event list approach, the simulation problem is posed using recurrence relations. We bring three algorithmic ideas to bear on parallel simulation: parallel prefix computation, parallel merging, and iterative folding. Effici ...

7 Implementing an efficient vector instruction set in a chip multi-processor using micro-threaded pipelines

Chris Jesshope

January 2001 **Australian Computer Science Communications , Proceedings of the 6th Australasian conference on Computer systems architecture**, Volume 23 Issue 4

Full text available: pdf(911.26 KB) Publisher Site Additional Information: full citation, abstract, references, citings

This paper looks at a combination of two techniques, one of which, using a vector instruction set, has a long history dating back to pipelined vector supercomputers, such as the Cray 1 and its successors. The other technique, multi-threading, is also well understood. The novel approach proposed in this paper combines both vertical and horizontal micro-threading with vector instruction descriptors. It will be shown that a family of threads can represent a

vector instruction with dependencies betw ...

8 Quantifying loop nest locality using SPEC'95 and the perfect benchmarks
Kathryn S. McKinley, Olivier Temam
November 1999 **ACM Transactions on Computer Systems (TOCS)**, Volume 17 Issue 4

Full text available: pdf(635.63 KB) Additional Information: full citation, abstract, references, citings, index terms

This article analyzes and quantifies the locality characteristics of numerical loop nests in order to suggest future directions for architecture and software cache optimizations. Since most programs spend the majority of their time in nests, the vast majority of cache optimization techniques target loop nests. In contrast, the locality characteristics that drive these optimizations are usually collected across the entire application rather than at the nest level. Researchers have studied nu ...

9 The fuzzy barrier: a mechanism for high speed synchronization of processors
Rajiv Gupta
April 1989 **ACM SIGARCH Computer Architecture News , Proceedings of the third international conference on Architectural support for programming languages and operating systems**, Volume 17 Issue 2

Full text available: pdf(1.12 MB) Additional Information: full citation, abstract, references, citings, index terms

Parallel programs are commonly written using barriers to synchronize parallel processes. Upon reaching a barrier, a processor must stall until all participating processors reach the barrier. A software implementation of the barrier mechanism using shared variables has two major drawbacks. Firstly, the execution of the barrier may be slow as it may not only require execution of several instructions and but also result in hot-spot accesses. Secondly, processors that are stalled waiting for ot ...

10 Unboundedly parallel simulations via recurrence relations for network and reliability problems
Albert G. Greenberg, Boris D. Lubachevsky, Isi Mitrani
December 1990 **Proceedings of the 22nd conference on Winter simulation**

Full text available: pdf(313.71 KB) Additional Information: full citation, references, citings, index terms

11 The Wisconsin Wind Tunnel: virtual prototyping of parallel computers
Steven K. Reinhardt, Mark D. Hill, James R. Larus, Alvin R. Lebeck, James C. Lewis, David A. Wood
June 1993 **ACM SIGMETRICS Performance Evaluation Review , Proceedings of the 1993 ACM SIGMETRICS conference on Measurement and modeling of computer systems**, Volume 21 Issue 1

Full text available: pdf(1.40 MB) Additional Information: full citation, references, citings, index terms

12 Superfast parallel discrete event simulations
Albert G. Greenberg, Boris D. Lubachevsky, Isi Mitrani
April 1996 **ACM Transactions on Modeling and Computer Simulation (TOMACS)**, Volume 6 Issue 2

Full text available: pdf(421.63 KB) Additional Information: full citation, abstract, references, index terms, review

Nonconventional parallel simulations methods are presented, wherein speed-ups are not limited by the number of simulated components. The methods capitalize on Chandy and

Sherman's space-time relaxation paradigm, and incorporate fast algorithms for solving recurrences. Special attention is paid to implementing these algorithms on currently available massively parallel SIMD computers. As examples, "superfast" simulations for open and closed queuing networks and for the slotted ALO ...

Keywords: fixed-point computations, massively parallel computations, recurrences, relaxation, superlinear speedup, unbounded parallelism

13 Parallel logic simulation of VLSI systems

Mary L. Bailey, Jack V. Briner, Roger D. Chamberlain
September 1994 **ACM Computing Surveys (CSUR)**, Volume 26 Issue 3

Full text available: pdf(3.74 MB) Additional Information: full citation, abstract, references, citings, index terms

Fast, efficient logic simulators are an essential tool in modern VLSI system design. Logic simulation is used extensively for design verification prior to fabrication, and as VLSI systems grow in size, the execution time required by simulation is becoming more and more significant. Faster logic simulators will have an appreciable economic impact, speeding time to market while ensuring more thorough system design testing. One approach to this problem is to utilize parallel processing, taking ...

Keywords: circuit structure, parallel architecture, parallelism, partitioning, synchronization algorithm, timing granularity

14 Compiling Fortran D for MIMD distributed-memory machines

Seema Hiranandani, Ken Kennedy, Chau-Wen Tseng
August 1992 **Communications of the ACM**, Volume 35 Issue 8

Full text available: pdf(5.38 MB) Additional Information: full citation, references, citings, index terms, review

Keywords: Fortran D, concurrent languages, distributed languages, distributed programming, parallel languages, parallel programming

15 An analysis of rollback-based simulation

Boris Lubachevsky, Adam Schwartz, Alan Weiss
April 1991 **ACM Transactions on Modeling and Computer Simulation (TOMACS)**, Volume 1 Issue 2

Full text available: pdf(2.47 MB) Additional Information: full citation, references, citings, index terms, review

16 Function unit specialization through code analysis

Daniel Benyamin, William H. Mangione-Smith
November 1999 **Proceedings of the 1999 IEEE/ACM international conference on Computer-aided design**

Full text available: pdf(105.29 KB) Additional Information: full citation, abstract, references, index terms

Many previous attempts at ASIP synthesis have employed template matching techniques to target function units to application code, or directly design new units to extract maximum performance. This paper presents an entirely new approach to specializing hardware for application specific needs. In our framework of a parameterized VLIW processor, we use a post-modulo scheduling analysis to reduce the allocated hardware resources while increasing the code's performance. Initial results i ...

17 Physical Experimentation with Prefetching Helper Threads on Intel's Hyper-Threaded Processors

Dongkeun Kim, Steve Shih-wei Liao, Perry H. Wang, Juan del Cuvillo, Xinmin Tian, Xiang Zou, Hong Wang, Donald Yeung, Milind Girkar, John P. Shen

March 2004 **Proceedings of the international symposium on Code generation and optimization: feedback-directed and runtime optimization**

Full text available: pdf(264.47 KB) Additional Information: full citation, abstract, citings

Pre-execution techniques have received much attention as aneffective way of prefetching cache blocks to tolerate the ever-increasingmemory latency. A number of pre-execution techniquesbased on hardware, compiler, or both have been proposed andstudied extensively by researchers. They report promising resultson simulators that model a Simultaneous Multithreading (SMT)processor. In this paper, we apply the helper threading idea ona real multithreaded machine, i.e., Intel Pentium 4 processor withHyp ...

18 Speculative precomputation: long-range prefetching of delinquent loads

Jamison D. Collins, Hong Wang, Dean M. Tullsen, Christopher Hughes, Yong-Fong Lee, Dan Lavery, John P. Shen

May 2001 **ACM SIGARCH Computer Architecture News , Proceedings of the 28th annual international symposium on Computer architecture**, Volume 29 Issue 2

Full text available: pdf(995.50 KB) Publisher Site Additional Information: full citation, abstract, references, citings, index terms

This paper explores Speculative Precomputation, a technique that uses idle thread context in a multithreaded architecture to improve performance of single-threaded applications. It attacks program stalls from data cache misses by pre-computing future memory accesses in available thread contexts, and prefetching these data. This technique is evaluated by simulating the performance of a research processor based on the Itanium™ ISA supporting Simultaneous Multithreading. Two primary for ...

19 Automatic loop transformations and parallelization for Java

Pedro V. Artigas, Manish Gupta, Samuel P. Midkiff, José E. Moreira

May 2000 **Proceedings of the 14th international conference on Supercomputing**

Full text available: pdf(969.06 KB) Additional Information: full citation, abstract, references, citings, index terms

From a software engineering perspective, the Java programming language provides an attractive platform for writing numerically intensive applications. A major drawback hampering its widespread adoption in this domain has been its poor performance on numerical codes. This paper describes a prototype Java compiler which demonstrates that it is possible to achieve performance levels approaching those of current state-of-the-art C, C++ and Fortran compilers on numerical codes. We describe a new ...

20 Overview of a high-performance programmable pipeline structure

Franc¸ois Bodin, Franc¸ois Charot, Charles Wagner

June 1986 **Proceedings of the 3rd international conference on Supercomputing**

Full text available: pdf(2.05 MB) Additional Information: full citation, abstract, references, index terms

This paper aims at describing a high-performance programmable pipeline architecture consisting of a linear array of PCS processors. The PCS processor which is capable of performing 20 million floating-point operations per second (20 MFLOPS) has been built from off-the-shelf chips on a wire-wrapped board. The prototype processor is attached to a SUN-3 workstation. Efficient microcode is generated using the microcode compiler that has been designed and implemented. The microcode op ...

Search Results Results **1 - 10** of about **10,100** for **"single instruction" multiple loop** - 0.33 sec. (About this page

1. Single Instruction **Multiple** Data
   News, trends and advice for CIOs and IT executives. ... whatis.com: searchCIO.com Definitions - **Single Instructio** succession of instructions (a **loop**) can now be performed in one instruction ...
   www.searchebusiness.com/sDefinition/0%2C%2Csid19_gci214206%2C00.html - 28k - Cached - More pages from t

2. Single Instruction **Multiple** Data
   News and advice for IT professionals working with Web services and application integration. ... whatis.com: searchW Data ... requires a repeated succession of instructions (a **loop**) can now be performed in one instruction ...
   searchwebservices.techtarget.com/sDefinition/0%2C%2Csid26_gci214206%2C 00.html - 31k - Cached - More page

3. What is Single Instruction **Multiple** Data - a definition from the leading computer technical encycl
   ... All Categories Software Applications. **Single Instruction Multiple** Data ... What usually requires a repeated succe instruction ...
   www.whatis.com/definition/0%2C%2Csid9_gci214206%2C00.html - 34k - Cached - More pages from this site

4. http://www.reed-electronics.com/ednmag/index.asp?layout=articlePrint&articleID=CA56716
   ... The program sequencer features internal **loop** counters and **loop** stacks, enabling looped code to execute ... 64-b features of this ...
   www.reed-electronics.com/ednmag/index.asp?layout=articlePrint&articleI D=CA567... - 95k - Cached - More pages f

5. Introduction to Parallel Programming
   ... SIMD Model: **Single Instruction**, **Multiple** Data Stream ... each processor executes its portion of the **loop** a num
   www.mhpcc.edu/training/workshop/parallel_intro/MAIN.html - 84k - Cached - More pages from this site

6. CSC4809 Parallel Architectures
   ... SISD - **Single-Instruction** Stream, Single Data Stream. SIMD - **Single-Instruction** Stream, **Multiple** Data Stream parallelism ...
   www.dimensional.com/~dazzler/CSC4809A.htm - 16k - Cached - More pages from this site

7. EDN - 2003 DSP directory (continued) - 4/3/2003 - EDN - CA288935
   ... including MIMD (**multiple**-instruction-**multiple**-data) and SIMD (**single-instruction-multiple**-data) instructions an levels of repeat ...
   www.reed-electronics.com/ednmag/index.asp?layout=article&stt=000&artic leid=CA... - More pages from this site

8. Architectures for DSP: The Options Multiply (PDF)
   ... • Enhanced conventional DSPs. • **Single-instruction**, **multiple**-data (SIMD ... (Computes one tap per **loop** iterati
   www.bdti.com/articles/options_sfsu.pdf - 685k - View as html - More pages from this site

9. Understanding the New DSP Processor Architectures (PDF)
   ... • Enhanced conventional DSPs. • **Single instruction**, **multiple** data (SIMD ... Compare to FIR filter inner **loop** co
   www.bdti.com/articles/understand_000407.PDF - 210k - View as html - More pages from this site

10. Analog Devices: ADSP-21160N: Product Page:High Performance 32-Bit SHARC DSP, 100 MHz
    ... Zero-Overhead Looping and Single-Cycle **Loop** Setup, Providing Efficient Program Sequencing ... 27 mm Metric Architecture provides two ...
    www.analog.com/Analog_Root/productPage/productHome/0%2C2121%2Cgeneric% 253DADS... - 35k - Cached - M

Results Page:

Web Results **1** - **10** of about **127** for **"single instruction" "multiple loop"**. (**1.45** seconds)

Vectorization with the Intel Compilers (Part I)
... today feature multimedia extensions that support SIMD (**single-instruction**-multiple-data ... these pragmas can be used to suggest **multiple loop**-distribution points ...
www.devx.com/Intel/Article/20093 - 39k - Sep 22, 2004 - Cached - Similar pages

[PDF] Implementing an efficient vector instruction set in a chip multi ...
File Format: PDF/Adobe Acrobat - View as HTML
... vector architecture, threads are used to execute **multiple loop** bodies simultaneously ... to 64 operations could be executed using a **single instruction**, the length ...
carol.science.uva.nl/~jesshope/Papers/acsac2001.pdf - Similar pages

[PDF] Real-Time High-Throughput Sonar Beamforming Kernels Using Native ...
File Format: PDF/Adobe Acrobat - View as HTML
... data words, and performs operations on multiple words with a **single instruction**. ... a technique to enlarge a program's basic blocks – **multiple loop** copies are ...
www.ece.utexas.edu/~allen/Asilomar99/Asilomar99.pdf - Similar pages

CIS 400 Lecture 11
... Scope of loop variable? 6) **Multiple loop** exits. ... Up until now, we have been covering... VonNewmann architecture, (**single instruction** at a time). ...
www.engin.umd.umich.edu/CIS/ course.des/cis400/maxim/lectures/lect10.htm - 51k - Cached - Similar pages

Citations: A processor architecture for Horizon - Thistle, Smith ...
... minimum of the assumed latencies across all operations within a **single instruction**. ... To do this, we allow **multiple loop** iterations to execute concurrenctly on a ...
citeseer.ist.psu.edu/context/85085/0 - 33k - Cached - Similar pages

[PDF] Efficiency of microSIMD architectures and index-mapped data for ...
File Format: PDF/Adobe Acrobat - View as HTML
... A SIMD (**Single Instruction** Multiple Data) architecture has the same datapaths as the MIMD architecture, except that a **single instruction** is issued to all the ...
palms.ee.princeton.edu/PALMSopen/lee99efficiency.pdf - Similar pages

Parallel Processing Concepts
... The "single" in **single-instruction** doesn't mean that there's only ... concurrency, parallelism and vectorization, view dependencies in **multiple-loop** situations, ie ...
www.tc.cornell.edu/Services/Edu/topics/ParProgCons/ - 87k - Cached - Similar pages

[PDF] Optimizing Applications with the Intel C++ and Fortran Compilers ...
File Format: PDF/Adobe Acrobat - View as HTML
... Vectorization Vectorization detects patterns of sequential data accesses by the same instruction, and transforms the code for **Single Instruction** Multiple Data ...
www.intel.com/software/products/compilers/ c60/techtopics/Compiler_Optimization_6.pdf - Similar pages

Emulation
... In the 601, a **single instruction** can extract and position the four-bit major ... compensated by the fact that it is not repeated during **multiple loop** executions. ...
www.mactech.com/articles/ mactech/Vol.10/10.09/Emulation/ - 60k - Sep 21, 2004 - Cached - Similar pages

[PDF] IA-64 and Itanium\(tm\) Processor Architecture Overview

File Format: PDF/Adobe Acrobat - View as HTML
... Architecture *Other brands and names are the property of their respective owners © Copyright 2002-2003 Intel Corporation **Single Instruction**, Multiple Data ...
www.cerfacs.fr/Seminars/2003/INTEL/Xeon_Itanium.pdf - Similar pages

Gooooooooogle ▶

Result Page: **1** 2 3 4 5 6 7 8 9 10 **Next**

Free! Get the Google Toolbar. Download Now - About Toolbar

"single instruction" "multiple loop" Search

Search within results | Language Tools | Search Tips | Dissatisfied? Help us improve

Google Home - Advertising Programs - Business Solutions - About Google